Recycled Equations Help Verify Livermore Codes

CODES that model complicated hydrodynamics are an important component of Livermore's stockpile stewardship efforts. A persistent concern for code developers and users is how well these codes model reality. Before scientists trust the results from a model, they must quantify the difference, or error, between the simulations it produces and physical reality.

Two kinds of errors may exist in the simulation: those due to code design and those due to code implementation. Design errors occur because the input parameters or the equations being solved do not accurately reflect the physical processes to be modeled. These errors result primarily because scientists do not have the detailed information needed to completely understand the physics. To address design errors, scientists must collect higher fidelity data, for example, by conducting experiments in laboratories. The resulting data help them determine which parameters to include and which equations the code must solve to accurately model the problem. This process of confirming that the code is solving the correct equations is called code validation.

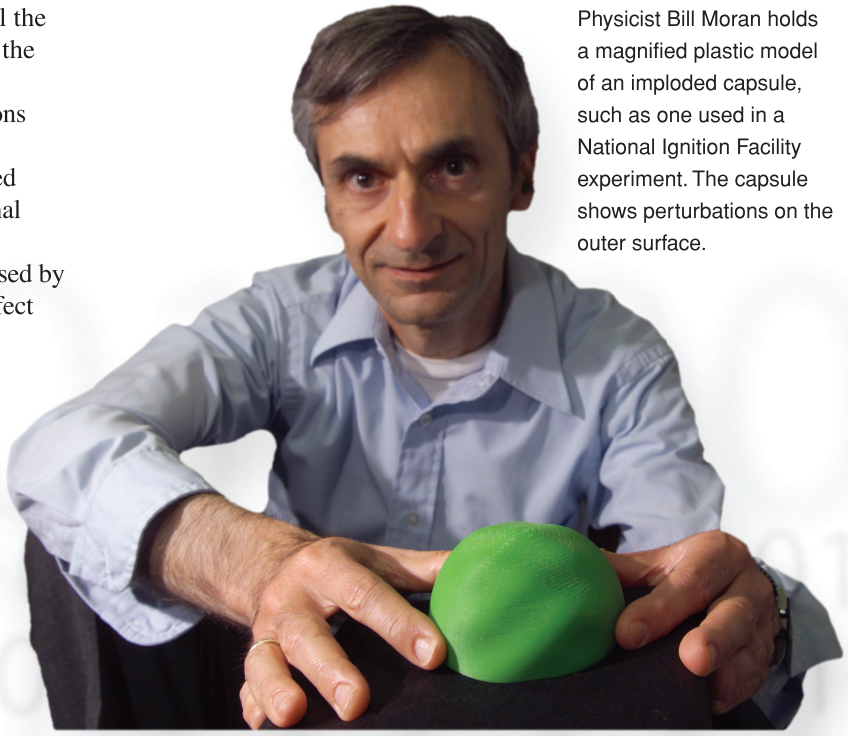
Scientists must also confirm that the code solves the equations correctly, a process called code verification. Implementation errors can sometimes occur because the equations are not solved correctly. A modern hydrodynamics code with three-dimensional (3D) capabilities has more than 500,000 lines of coding and 5,000 modules. Any mistake or bug in the coding, whether caused by a typographic error or insufficient computing resources, can affect the model's accuracy.

Use of Test Problems

One reliable approach to verification is to have a code run test problems with known answers. These problems are simple enough to be solved analytically—that is, by working through differential equations and other mathematical expressions to determine the exact answer to each problem. The computer code also processes these test problems. Then by comparing the known solution with the code-generated one, scientists can determine code error for these problems.

Over the years, Livermore researchers have developed sets of test problems for verifying the implementation in hydrodynamics codes, such as those used to study the implosion and explosion of a capsule in a National Ignition Facility (NIF) experiment. However, test problems have limitations as a verification method because they do not adequately represent all of the physical processes that occur in a NIF capsule. They also do not, in general, account for the 3D nature of the implosion. For example, in a direct drive experiment, the finite number of laser beams impinging on a capsule introduces perturbations. These perturbations can alter the implosion and affect the capsule's performance. Those effects, however, will not appear in simulations of typical one-dimensional test problems because of the simplified physics. Thus, when scientists compare results of the test problems with more complex simulations of imploding capsules, they cannot accurately quantify the uncertainties in the code's calculations.

To address this problem of code verification, Livermore physicist Bill Moran has developed new test problems that have analytic solutions for an imploding shell under exponentially decaying pressure. "The test problems are analogous to the hydrodynamics of an imploding capsule, such as inertial confinement fusion (ICF) targets," says Moran. "Although the solutions are still simplified, they can include 3D effects in a way that can be checked exactly."



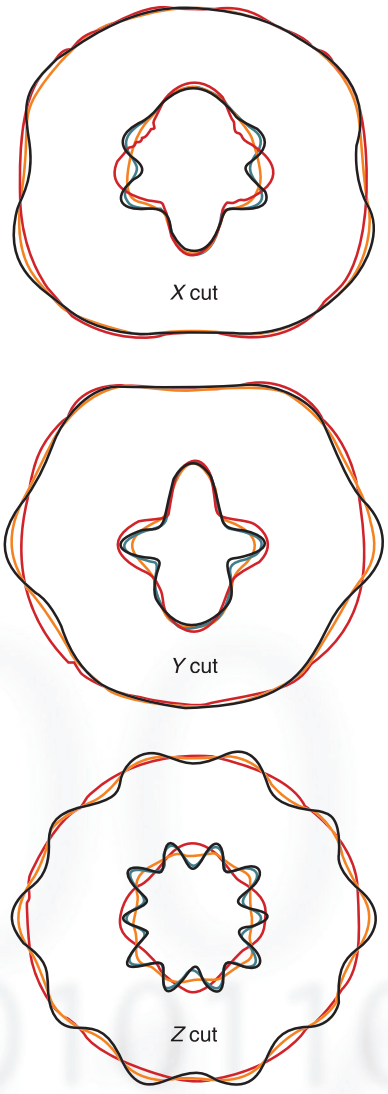
Physicist Bill Moran holds a magnified plastic model of an imploded capsule, such as one used in a National Ignition Facility experiment. The capsule shows perturbations on the outer surface.

The test problems provide an independent check of the 3D codes developed for the National Nuclear Security Administration's (NNSA's) Advanced Simulation and Computing (ASC) Program. They also help scientists compare the efficiency of the algorithms and codes used in NIF capsule implosion simulations.

Analyze This

Moran's solutions build on a previous assignment, in which he studied the containment of underground nuclear experiments for the Laboratory's former Nuclear Test Program. This containment research focused on geophysical applications and seismic decoupling. Moran realized he could "recycle" those past analyses by turning the problems inside out to model an imploding shell. In the containment problem, a nuclear device explodes in an underground cavity, and the resulting pressure is a force on the inside pressing out on a surface. In a NIF capsule, the impinging laser beams ablate the outer surface and create a force on the outside pressing in on a shell—the NIF capsule.

To investigate how well new codes model the hydrodynamics of imploding capsules, Moran developed test problems for three kinds of shells. The first kind is an incompressible fluid shell. This shell deforms easily but maintains its volume, much like a balloon filled with water. The balloon can be squeezed and deformed, but the volume of water inside it does not change. The second is a compressible shell filled with gas. This shell is similar to a pillow that, if compressed uniformly, can be deformed and reduced in volume. The third is also an incompressible fluid shell, but it has minor perturbations (small bumps) on both the inner and outer surfaces.



One test problem compared the cavity shape of an imploded inertial confinement fusion shell. The comparisons view the shape across three planes: (a) a horizontal, or *X*, cut; (b) a vertical, or *Y*, cut; and (c) an axial, or *Z*, cut. The black curve shows the known solution for the test problem, red is from a simulation with two-degree zoning, blue is with one-degree zoning, and green is with one-half-degree zoning. Note that, in places, the two-degree calculation is out of phase with the known solution, showing the importance of zoning appropriately.

Getting Closer to Reality

Moran added several realistic features in the test problems, so the solutions more closely reflect what happens in a NIF capsule. For example, he includes the initial shell thickness of a typical capsule, the initial inner radius, and the initial density of the capsule. He also incorporates a mathematical form of the applied pressure, chosen to be close to the pressure experienced by an imploding capsule.

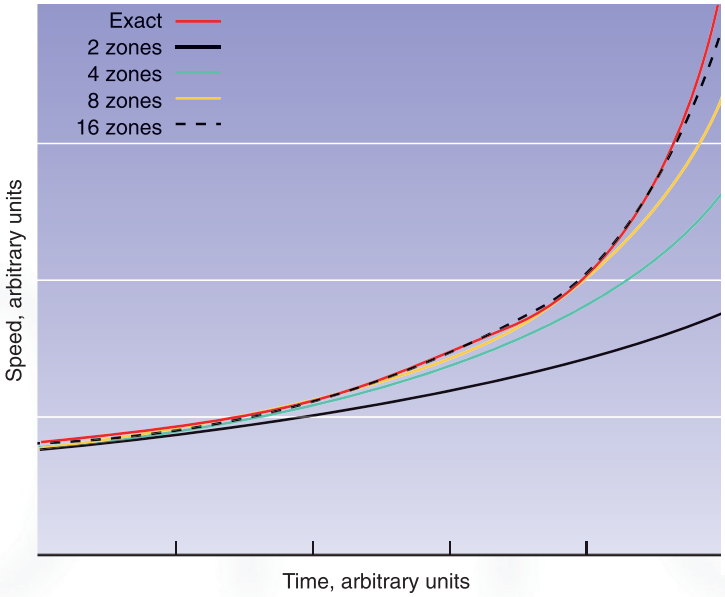
Solving for a shell that has minor perturbations on its inner and outer surface was another way to bring the analytic approach closer to the geometry of a real capsule. For example, during capsule manufacturing, small surface perturbations are always present. Assessing how accurately a code simulates these small deviations is important to ensure that numerical artifacts in the code do not drown the small signal being measured. "The size of mesh we choose—whether we use more zones or fewer zones to model a shell—can determine whether we see the signal," says Moran. "Comparing the known analytic solutions with the code's results can help us determine what the trade-off is between the cost of running a simulation and its accuracy." Code users often consider this trade-off because increasing the accuracy requires a mesh with more zones and smaller mesh sizes, making the code run longer.

Putting Codes through Their Paces

In one test problem, Moran considered an incompressible shell with a representative pressure on the outer surface of the shell and analytically solved for the jump-off velocity—the velocity of the inner surface at the time pressure first affects it. Determining

the jump-off velocity provides a test of a code's ability to model shock propagation. The known solution to this problem was then compared to code calculations, and the error quantified for various mesh resolutions. Next, the analytic time evolution of the speed at the inner surface of the shell was compared to code calculations. Moran also worked out analytic solutions for pressure, strain rate, and strain across the shell thickness for various initial configurations.

To verify the 3D capabilities of the ASC codes, Moran adapted into a test problem the work on perturbation growth of Livermore physicist Karnig Mikaelian. "A practical concern we have in achieving a symmetric implosion is whether tolerances are more critical on one surface for reducing perturbations," says Moran. "So we run calculations that simulate the perturbation on one surface at a time. The results for thin shells indicate that both surfaces are critical. Having only one surface perfectly smooth won't stop the perturbations from feeding through to the other surface." Moran notes that these calculations model the cavity as a void without gas. "A more realistic representation would have the gas pushing back on the inner surface as the shell implodes. When a light material—the gas, in this case—pushes on a denser material—the shell—the surface becomes more susceptible to instability growth."



Code simulations of an incompressible fluid shell show the convergence toward the exact solution with finer meshes. When the number of mesh zones is reduced, the error in the calculation increases rapidly.

Moran compared the exact solution of the cavity hot spot with 3D simulations using different resolutions, including two-degree, one-degree, and one-half-degree zoning. The result: the smaller the zones, the better. "In fact," he says, "in the simulation with the coarse two-degree zoning, the amplitude of the instabilities is out of phase—has the opposite sign—at some locations compared with the exact result." Moran also made solid models of the cavity from the two-degree calculation using a rapid-prototype machine, which takes 3D code results and turns them into plastic models. The plastic models showed that, for two-degree calculations, the mesh is imprinted on the cavity surface, effectively drowning out small details.

Analytic solutions also can be used to evaluate the efficiency of various algorithms and codes by comparing the computer run time required to achieve a given accuracy in solving a specific problem. For instance, with a code running on 64 processors of the ASC Frost supercomputer, Moran modeled half of a spherical shell using one-half-degree zoning and various radial resolutions. In Lagrangian mode, the code required 80 radial zones and a processing time of 20,000 seconds (about 5.5 hours) to calculate the peak pressure with an accuracy of 0.5 percent. In arbitrary Lagrange–Eulerian mode, where the nodes were evenly distributed in the radial direction every cycle, the code achieved the same accuracy with only 24 radial zones in 10,000 seconds (about 2.75 hours).

Solutions Are Just the Start

Lawrence Livermore scientists and their colleagues at other locations are creating new test problems building on Moran's research. These problems will be added to existing ones, forming an enlarged suite for scientists to use.

Cynthia Nitta, who formerly led Livermore's verification and validation effort, notes that Moran's work is a breakthrough in code verification, particularly for 3D codes. "Bill's work represents important 3D verification analysis in imploding geometries for code simulation capabilities. The results on the effects of mesh size also provide critical information for evaluating future computer platform capability and capacity requirements."

—Ann Parker

Key Words: Advanced Simulation and Computing (ASC) Program, analytic solutions, code verification and validation, hydrodynamics, stockpile stewardship, three-dimensional (3D) modeling.

For further information contact Bill Moran (925) 422-7250 (moran1@llnl.gov).